homework 2 report

Algorithm and Optimizations

The algorithm I wrote is all contained within one file `apriori.java`. I divided up the algorithm into multiple different methods: 3 methods to scan the input file and write the output file, 1 method to sort the itemsets in descending order of their support counts, and 7 methods for the apriori algorithm itself. I created separate methods to find the set of frequent 1-itemsets, then to generate frequent k-itemset candidates from the $(k-1)$-itemsets, to generate subsets from the candidates, and then to prune candidates by extracting the frequent itemsets. I utilized HashMaps to keep track of the sets of frequent k-itemsets and the support counts of each itemset.

For my candidate generation method (`getCandidates()`), I utilized the $F_{k-1} \times F_{k-1}$ method. In this method, I sorted each frequent set of keywords (of length $k-1$) in alphabetical order and then merged them together if their first $(k-2)$ items were the identical. I made sure to check that the sets were not *completely* identical before merging to avoid duplicates.

Frequency Count Threshold and Minimum Support

To generate my `output.txt` file, I chose to use a minimum support value of 600. After testing out the results of different minimum support counts, I felt that this number was the best balance between getting a "high" enough number of frequent itemsets for analysis (generating 91 outputs) while filtering out more insignificant words like "lol". Subjectively, the results produced a decently wide range of frequent itemsets that would be useful to examine, from 1-itemsets up to 4-itemsets, with a diverse set of words that can still be clearly recognized as relating to the topic of the dataset tweets: the flu and getting flu shots.

Results

In the set of frequent itemsets generated, "flu", "shot", and "flu shot" were the top keyword groups, confirming that the topic of the tweets in the dataset were related to flu shots. Additionally, variations of "get flu shot" and "getting flu shot" were among the most popular word sets, revealing that the tweets may have frequently described the Twitter users' experiences or thoughts about receiving the flu shot. There were also many groups including "flu" and "shot" along with a description of time, such as "today," "time," "year," and "last." This suggests that many tweets also described *when* the authors may have received their flu shots or when they intended to receive it. Because "year" was among the frequent keywords, it's probable that a good number of tweets referenced the yearly nature of getting flu shots or the act of getting *this* year's shot.

A final interesting characteristic of the set of frequent itemsets is the co-occurrence of "sore" and "sick" with "flu shot." These could point to Twitter users' fears of getting the flu and/or facing negative symptoms due to getting the flu shot. For instance, since many people feel soreness in their bodies and sometimes even flu symptoms after receiving the shot, this may be an area of concern that people often talk about. This observation could be useful in an analysis about people's attitudes toward the getting flu shot––including why they decide to get it and feel about the symptoms afterward. This analysis could provide insight into vaccine distribution and general attitudes toward the vaccine.

Lessons and Conclusion

I had a valuable experience working on this assignment because it gave me hands-on experience with a frequent pattern mining algorithm. By actually implementing the algorithm, I gained a much better understanding of how it works and *why* it works. I initially felt overwhelmed by the large amount of data that I was given, but walking through the steps of parsing through all of the information to retrieve the important parts helped me see why data mining is so important and how effective it is at producing patterns and observations out of seemingly nothing. Additionally, this assignment helped improve my programming skills because it made me think about how to interpret pseudocode into my own program and break down an algorithm into smaller and smaller parts that I could implement through code.

Overall, this project made the data mining process feel much more "real" and relevant to me and gave me confidence that I will be able to use these skills and knowledge to perform more data analysis on my own in the future.